Perl 函数中 Perl 进程控制函数用法解析

本文和大家重点讨论一下 Perl 进程处理函数的用法，主要包括进程启动函数，进程终止函数和进程控制函数等内容，相信通过本文的学习你对 Perl 进程处理函数的用法一定会有深刻的认识。

1、进程启动函数

Perl 函数名 eval

调用语法 eval(string)

解说将 string 看作 Perl 函数名语句执行。

正确执行后，系统变量$@为空串，如果有错误，$@中为错误信息。

例子$print="print(\"hello,world\\n\");";

eval($print);

结果输出 hello,world

Perl 函数名 system

调用语法 system(list)

解说 list 中第一个元素为程序名，其余为参数。

system 启动一个进程运行程序并等待其结束,程序结束后错误代码左移八位成为返回值。

例子@proglist=("echo","hello,world!");

system(@proglist);

结果输出 hello,world!

Perl 函数名 fork

调用语法 procid=fork();

解说创建程序的两个拷贝--父进程和子进程--同时运行。子进程返回零，父进程返回非零

值，此值为子程序的进程 ID 号。

例子

```
1. $retval=fork();
2. if($retval==0){
3. #thisisthechildprocess
4. exit;#thisterminatesthechildprocess
5. }else{
6. #thisistheparentprocess
7. }
8.
```

结果输出无

Perl 函数名 pipe

调用语法 pipe(infile,outfile);

解说与 fork 合用，给父进程和子进程提供通信的方式。送到 outfile 文件变量的信息可以

通过 infile 文件变量读取。步骤：

1、调用 pipe

2、用 fork 将程序分成父进程和子进程

3、一个进程关掉 infile，另一个关掉 outfile

例子

```
1. pipe(INPUT,OUTPUT);
2. $retval=fork();
3. if($retval!=0){
4. #thisistheparentprocess
```

```
close(INPUT);

print("Enteralineofinput:\n");
$line=<STDIN>;
printOUTPUT($line);
}else{
#thisisthechildprocess
close(OUTPUT);
$line=<INPUT>;
print($line);
exit(0);
}
```

结果输出$

```
program
Enteralineofinput:
Hereisatestline
Hereisatestline
$
```

Perl 函数名 exec

调用语法 exec(list);

解说与 system 类似，区别是启动新进程前结束当前程序。常与 fork 合用，当 fork 分成两个

进程后，子进程用 exec 启动另一个程序。

例子

结果输出

Perl 函数名 syscall

调用语法 syscall(list);

解说调用系统 Perl 函数名，list 第一个元素是系统调用名，其余为参数。

如果参数是数字，就转化成 C 的整型数(typeint)。否则传递字符串的指针。详见 UNIX 的帮助 Perl 函数名文档。

使用 syscall 必须包含文件 syscall.pl，即：

require("syscall.ph");

例子

结果输出

2、进程终止函数

Perl 函数名 die

调用语法 die(message);

解说终止程序并向 STDERR 输出错误信息。message 可以为字符串或列表。如果最后一个参
数不包含换行符，则程序文件名和行号也被输出。

例子 die("Cannotopeninputfile");

结果输出 Cannotopeninputfileatmyprogline6.

Perl 函数名 warn

调用语法 warn(message);

解说与 die 类似，区别是不终止程序。

例子 warn("Danger!Danger!\n");

结果输出 Danger!Danger!

Perl 函数名 exit

调用语法 exit(retcode);

解说终止程序并指定返回值。

例子 exit(2);

结果输出无

Perl 函数名 kill

调用语法 kill(signal,proclist);

解说给一组进程发送信号。

signal 是发送的数字信号，9 为杀掉进程。

proclist 是进程 ID 列表。详见 kill 的 UNIX 帮助。

例子

结果输出

3、进程控制函数

Perl 函数名 sleep

调用语法 sleep(time);

解说将程序暂停一段时间。time 是停止的秒数。返回值为实际停止的秒数。

例子 sleep(5);

结果输出无

Perl 函数名 wait

调用语法 procid=wait();

解说暂停程序执行，等待子进程终止。

不需要参数，返回值为子进程 ID，如果没有子进程，返回-1。

例子

结果输出

Perl 函数名 waitpid

调用语法 waitpid(procid,waitflag);

解说暂停程序执行，等待特定的子进程终止。procid 为等待的进程 ID

例子

```
1. $procid=fork();
2. if($procid==0){
3. #thisisthechildprocess
4. print("thislineisprintedfirst\n");
5. exit(0);
6.
7. }else{
8. #thisistheparentprocess
9. waitpid($procid,0);
10. print("thislineisprintedlast\n");
11. }
12.
```

结果输出$program

thislineisprintedfirst

thislineisprintedlast

```
$
```

4、其它控制函数

Perl 函数名 caller

调用语法 subinfo=caller();

解说返回调用者的程序名和行号，用于 Perl 函数名 Debugger。

返回值为三元素的列表：

1、调用处的包名

2、调用者文件名

3、调用处的行号

例子

结果输出

Perl 函数名 chroot

调用语法 chroot(dir);

解说改变程序的根目录，详见 chroot 帮助。

例子

结果输出

Perl 函数名 local

调用语法 local($variable);

解说在语句块(由大括号包围的语句集合)中定义局域变量，仅在此语句块中起作用，对其

的改变不对块外同名变量造成影响。

千万不要在循环中使用，否则每次循环都定义一个新的局域变量！

例子

结果输出

Perl 函数名 times

调用语法 timelist=times

解说返回该程序及所有子进程消耗的工作时间。

返回值为四个浮点数的列表：

1、程序耗用的用户时间

2、程序耗用的系统时间

3、子进程耗用的用户时间

4、子进程耗用的系统时间

例子

结果输出